

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平5-257811

(43)公開日 平成5年(1993)10月8日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 12/10  
9/46

識別記号

3 4 0 F 8120-5B

庁内整理番号

H 7232-5B

F I

技術表示箇所

審査請求 未請求 請求項の数8(全 24 頁)

(21)出願番号 特願平5-3937

(22)出願日 平成5年(1993)1月13日

(31)優先権主張番号 特願平4-26040

(32)優先日 平4(1992)1月16日

(33)優先権主張国 日本(JP)

(71)出願人 000003078

株式会社東芝

神奈川県川崎市幸区堀川町72番地

(72)発明者 斎藤 光男

神奈川県川崎市幸区小向東芝町1番地 株  
式会社東芝研究開発センター内

(72)発明者 浅野 滋博

神奈川県川崎市幸区小向東芝町1番地 株  
式会社東芝研究開発センター内

(72)発明者 申 承昊

神奈川県川崎市幸区小向東芝町1番地 株  
式会社東芝研究開発センター内

(74)代理人 弁理士 則近 憲佑

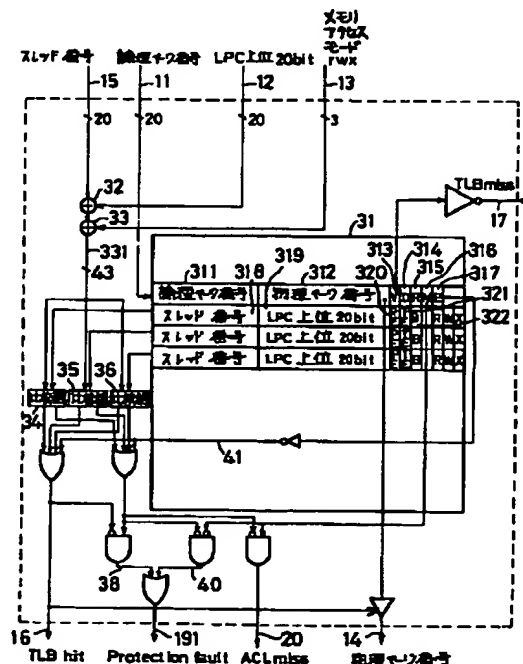
最終頁に続く

(54)【発明の名称】 メモリ管理装置

(57)【要約】

【目的】 複数の領域に分割され、その中を複数のスレッドが実行するアドレス空間の管理を行ない、メモリアクセスに対してアクセス権の制御を適切に行なう。

【構成】 論理アドレスから物理アドレスへの変換を行なう機能を有するもので、TLB(ページテーブルのキャッシュ)31中に、論理アドレスに対するアクセスを許すスレッドの番号と領域の番号と許すアクセスの方法を組にしたものであるアクセスコントロールリストを複数格納し、これらのアクセスコントロールリストのスレッド番号と領域番号の組のうち、現在実行中のスレッドの番号15とそのスレッドが現在実行しているプログラムの存在している領域の番号12を組み合わせたものに一致するものを比較器34~36で検出し、ここで一致するものが検出されるとTLBヒット16として出力するようにしている。



## 【特許請求の範囲】

【請求項1】 複数のスレッドを起動して仮想空間に割り付けられたプログラムを並行して実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうメモリ管理装置であって、

現在プログラムを実行しているスレッドの番号を記憶する手段と、

前記論理アドレスをどのスレッドがアクセス可能かを示す情報を記憶する手段と、

前記スレッド番号を持つスレッドが前記論理アドレスへアクセスしようとする場合に前記アクセス可能かを示す情報を用いて前記アクセスをしようとしているスレッドにアクセスの権限があるか否かを検証する検証手段とを具備し、

この検証手段によりアクセスの正当性が保証された場合に前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とするメモリ管理装置。

【請求項2】 検証手段において、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号の一致を調べる際に、スレッド番号の全部又は一部をマスクする手段を具備することを特徴とする請求項1記載のメモリ管理装置。

【請求項3】 検証手段において、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号について所定の論理演算を行い、論理演算結果が真であった場合にスレッド番号が一致したと見做す手段を具備することを特徴とする請求項1記載のメモリ管理装置。

【請求項4】 複数の領域に分割された仮想空間に割り付けられたプログラムを実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうメモリ管理装置であって、

前記論理アドレスをアクセスするプログラムの置かれている領域がどれであるかを検知する手段と、

前記論理アドレスが前記仮想空間中のどの領域に置かれているプログラムからアクセス可能かを表す情報を記憶する手段と、

プログラム実行中に前記論理アドレスへのアクセス命令が出現した場合に前記アクセス可能かを表す情報を用いて前記アクセス命令が置かれている領域からのアクセスが正当か否かを検証する検証手段とを具備し、

この検証手段によりアクセスの正当性が保証された場合に、前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とするメモリ管理装置。

【請求項5】 検証手段において、どの領域からのアクセスが可能かを示す情報のなかの領域番号と、アクセスしようとしているプログラムの置かれている領域の領域番号の一致を調べる際に、領域番号の全部又は一部をマスクする手段を具備することを特徴とする請求項4記載

のメモリ管理装置。

【請求項6】 検証手段において、どの領域からのアクセスが可能かを示す情報のなかの領域番号と、アクセスしようとしているプログラムの置かれている領域の領域番号について所定の論理演算を行い、論理演算結果が真であった場合に領域番号が一致したと見做す手段を具備することを特徴とする請求項4記載のメモリ管理装置。

【請求項7】 仮想空間に割り付けられたプログラムを実行する計算機に設けられ、アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段を有するメモリ管理装置であって、

前記アドレス変換テーブル中のアドレス対の各々に対して複数のアクセス保護情報を記憶する手段と、

前記複数のアクセス保護情報を参照し、実行中のプログラムの仮想空間内の位置からのあるいはプログラムを実行中の実行主体による前記指示されたアドレスへのアクセスが許可されるか否かを検証する手段と、

前記検証結果を出力する手段とを具備したことを特徴とするメモリ管理装置。

【請求項8】 仮想空間に割り付けられたプログラムを実行する計算機に設けられ、

アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段と、

前記アドレス変換テーブルの一部を置くキャッシュメモリを有するメモリ管理装置であって、

前記アドレス変換テーブル中に設けられ、アドレス対の各々に対して複数の可変個のアクセス保護情報を記憶する第1の記憶手段と、

前記キャッシュメモリ中に設けられ、アドレス対に各々に対して複数の固定個のアクセス保護情報を記憶する第2の記憶手段と、

前記第2の記憶手段中の複数のアクセス保護情報を検証する検証手段と、

前記検証手段で検証できなかった場合に、前記キャッシュメモリ中の第2の記憶手段の内容のみを、前記アドレス変換テーブル中の第1の記憶手段の内容の一部と入れ換える手段を具備したことを特徴とするメモリ管理装置。

## 【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、メモリアccessを制御するメモリ管理装置に関するものである。

【0002】

【従来の技術】近年、ネットワーク技術の発達や並列計算機技術の発展により、例えばサーバ・クライアント型プログラミングのように、複数のプログラムがデータを

## 3

共有し、協調しあって処理を進める方式が広がって来た。

【0003】この方式では、それぞれのプログラムは、仮想記憶の技術を用いて別々のアドレス空間におかれているため、データを共有するためにはオペレーティングシステム(OS)を介してデータのやりとりをしなければならず、OSのオーバーヘッドにより処理速度が遅くなってしまうという欠点があった。

【0004】そこで、一つのアドレス空間内に、実行主体であるスレッドを複数走らせ、スレッド間のデータの共有を、OSのオーバーヘッドなしでおこなうという方式もおこなわれている。

【0005】この方式は、もともとスレッド間の保護を考えていないものであるため、あるスレッドが使っているデータを他のスレッドから保護しようすると、オーバーヘッドの大きい方法を探らなければならない。

【0006】すなわち、アドレス空間ごとの保護機能を利用してスレッド毎に別々のアドレス空間を割り当て、それぞれのアドレス空間に、そのスレッドからアクセスできる部分だけを置くことになる。これはかなりオーバーヘッドが大きく実用的ではない。以上が第1の問題点、つまり1つのプログラム中でスレッドごとに許されるアクセスの種類を変えることができないという問題点である。

【0007】さらにスレッド毎にアドレス空間を作るためには、それぞれの仮想空間ごとにページテーブルを用意しなければならず、同じプログラムは別々のアドレス空間内でも同じ位置に置かなければならないが、こうすると複数のスレッドで共通にアクセスできる領域についても、アクセスが許されるスレッド全てが、同じ論理-物理アドレスの変換情報を重複してページテーブル中に持たなければならないため、ページテーブルとして使うメモリが無駄である。

【0008】もちろん、複数のスレッドで共通にアクセスできる領域の、許されるアクセス方法が同じであれば、それらのスレッドで、一つのページテーブルの部分を共有することは出来、ページテーブルとして使うメモリの無駄をなくすことが出来るが、この場合でもTLB(Translation Look-aside Buffer)のようなページテーブルのキャッシュを使っている場合には、無駄が発生し、アドレス変換の平均速度を低下させる。

【0009】その理由を説明すると、まず、ページテーブルのキャッシュの1エントリは、ページテーブルの情報とスレッド番号を組にして持っているため、スレッドが違えば、ページテーブルの情報が全く同じでも、異なるキャッシュエントリを使うことになる。

【0010】したがって、同じ論理-物理アドレス対をなすが、スレッド番号の異なるものがキャッシュの複数のエントリを占有してしまうことになり、ページテーブルのキャッシュ内に格納できる物理アドレスの種類が少

## 4

なくなってしまう、キャッシュの大きさが小さくなったのと同様に、アドレス変換の速度の低下が起こる。

【0011】さらに、複数のページテーブルに置かれた同じページに対する情報を管理しなければならないため、ページングの処理が複雑になるという問題点がある。例えば、複数のスレッドが同じ物理ページをそれぞれの仮想アドレス空間にマッピングしている時に、そのページをページアウトしようすると、その物理ページをマッピングしている全ての仮想空間を探して、そのページに対応するページテーブルのエントリを無効化する処理が必要になる。

【0012】また、ページインする時も、ページインしようとしている物理ページを共有する全てのアドレス空間の、そのページに対応するページテーブルのエントリを直さなければならない。これらの処理時間が平均アドレス変換時間に加算される。以上が第2の問題点、つまり、ページテーブルや、ページテーブルのキャッシュのメモリが無駄になり、かつ、管理が複雑になり非効率であるという問題点である。

【0013】さらに、別の方法として一つのアドレス空間を複数の領域に分けて、スレッドがどの領域のプログラムを実行しているかによって、そのスレッドの持つ権限を変える、ということが、アプリケーションによっては有効である。例えば、領域Aのデータを変えられるのは、別のある領域Bにあるプログラムを実行しているスレッドに限る(領域Cにあるプログラムを実行しているスレッドは領域Aのデータを変えられない)ようにすれば、領域Aのデータを、一定の条件を保ちながら変更することが可能となる。領域Aにデータベースのデータ、領域Bにデータベースのプログラムでそのデータへのアクセスルーチンを入れておけば、データベース処理に関して、データベースのデータを保護しつつ、且つ、OSを介することなく、そのデータをアクセスできるため、高速なアクセスが可能となる。

【0014】しかし、従来のメモリ管理装置(例えばインテル社の80486プロセッサ)では、一つのスレッドが実行するプログラムの範囲をセグメント方式等を利用して異なるアクセスレベルに移ることによってアクセス権限を変えることは出来たが、アクセスレベルは、順序関係を持ち、レベルが上がった場合にはそのレベル以下でアクセス可能な領域は全てアクセスできるというものであり、かつハードウェアによってレベルの数を限定されてしまうため、アプリケーションプログラムとOSとカーネルといったような順序関係をもったアクセス権の制御しか利用できないなど柔軟性に欠けるという欠点があった。

【0015】以上が第三の問題点、つまり一つのアドレス空間中に複数の領域に分けてプログラムやデータ等を配置した場合、領域毎に許されるアクセスの種類を柔軟

に変わることができないという問題点である。

【0016】

【発明が解決しようとする課題】このように、従来のメモリ管理装置で、一つのアドレス空間内を実行する複数の実行主体（スレッド）の間で保護を行おうとすると、同じ物理ページをマッピングしているアドレス空間が多数あるため、ページングを行なう場合には、全てのアドレス空間を探して、そのページがマッピングされているものを見つけ、それらに対してページングの処理を行わなければならない等、オーバーヘッドが大きいという第1の問題点があった。

【0017】さらに、同じ物理アドレスに対してページテーブル中の多くのエントリを占有するために、他の物理アドレスに対するアドレス変換を阻害することになり、また、ページテーブルのエントリが複数のプログラムから共有されているか否かに関わらず無効化されてしまい、エントリを満たすための不必要な処理が行なわれる欠点があった。また、複数のプログラムが持つアクセスの権利の正しさを同時に検査しなければならない場合に、係る検査にかかる処理が複雑になってしまうという欠点があった。つまり、ページテーブルやページテーブルキャッシュに同じ情報が複数置かれてしまい、メモリが無駄になり、キャッシュの効果が薄れ、非効率であるという第2の問題点があった。

【0018】また、一つのアドレス空間を複数の領域に分けて、スレッドがどの領域のプログラムを実行しているかによって、そのスレッドの権限を変える、ということは、従来のメモリ管理装置では、柔軟に行なえないという第3の問題点があった。

【0019】本発明は、上記事情に鑑みなされたもので、複数のプログラムで共有されるアドレス空間の管理を行ない、メモリアクセスに対してアクセス権限の制御を適切に行なうことが出来、しかもアドレス変換時間を延長することもないメモリ管理装置を提供することを目的とする。

【0020】

【課題を解決するための手段】上記第一の問題点を解決する本発明に係るメモリ管理装置は、複数のスレッド（実行主体）を起動して仮想空間に割り付けられたプログラムを並行して実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうものであって、現在プログラムを実行しているスレッドの番号を記憶する手段と、前記論理アドレスをどのスレッドがアクセス可能かを示す情報を記憶する手段と、前記スレッド番号を持つスレッドが前記論理アドレスへアクセスしようとする場合に前記アクセス可能かを示す情報を用いて前記アクセスをしようとしているスレッドにアクセスの権限があるか否かを検証する検証手段とを具備し、この検証手段によりアクセスの正当性が保証された場合に前記論理アドレスから物理アドレスへの変換結果を出力するこ

とを特徴とする。

【0021】上記第二の問題点を解決する本発明に係るメモリ管理装置は、仮想空間に割り付けられたプログラムを実行する計算機に設けられ、アドレス変換テーブルに保持されている複数のアドレスに対して指示されたアドレスとの一致を検出し、該一致が検出された際に前記指示されたアドレスと対になるアドレスを出力する手段を有するものであって、前記アドレス変換テーブル中のアドレス対の各々に対して複数のアクセス保護情報を記憶する手段と、前記複数のアクセス保護情報を参照し、実行中のプログラムの仮想空間内の位置からあるいはプログラムを実行中のスレッド（実行主体）による前記指示されたアドレスへのアクセスが許可されるか否かを検証する手段と、前記検証結果を出力する手段とを具備したことを特徴とする。

【0022】上記第三の問題点を解決する本発明に係るメモリ管理装置は、複数の領域に分割された仮想空間に割り付けられたプログラムを実行する計算機に設けられ、論理アドレスから物理アドレスへの変換を行なうものであって、前記論理アドレスをアクセスするプログラムの置かれている領域がどれであるかを検知する手段と、前記論理アドレスが前記仮想空間中のどの領域に置かれているプログラムからアクセス可能かを表す情報を記憶する手段と、プログラム実行中に前記論理アドレスへのアクセス命令が出現した場合に前記アクセス可能かを表す情報を用いて前記アクセス命令が置かれている領域からのアクセスが正当か否かを検証する検証手段とを具備し、この検証手段によりアクセスの正当性が保証された場合に、前記論理アドレスから物理アドレスへの変換結果を出力することを特徴とする。

【0023】

【作用】第1の発明によれば、メモリ管理装置中にアドレス空間中の各領域（位置）に対してスレッドごとのアクセス可否を示す情報があり、更に現在実行中のスレッドを識別できるので、現在実行中のスレッドがアクセス先の領域（位置）にアクセスできる権限があるかの検証を行うことで、同じアドレス空間中の異なるスレッドごとのアクセス制御が可能となる。

【0024】ここで、スレッドとは実行主体であり、CPUの割り当て単位である。つまりOSはスレッド番号ごとにCPUのレジスタ等のメモリ以外に現在の実行を制御する情報を管理する。OS中にはこれらの情報を回避するメモリ領域があり、スレッドを切り替える時はこの領域の内容とCPUのレジスタ等とを入れかえることで実行の切替えを実現する。コンテキストとも呼ばれる。

【0025】第2の発明によれば、メモリ管理装置中のページテーブル（論理アドレスと物理アドレスの対応表）の各エントリに対応して複数のアクセス保護情報を保持することができるので、従来の方式のように単一の

## 7

アクセス情報しか保持できないもので構成するのと比較してページテーブル及びそのキャッシュのメモリ使用量が大幅に減り、効率が向上する。

【0026】第3の発明によれば、メモリ管理装置中にアドレス空間の各領域（位置）に対してどの領域からアクセスできるかを示す情報があり、更に現在実行中の領域を識別できるので、現在実行中のプログラムがアクセス先の領域（位置）にアクセスできる権限があるかの検証が行え、同じアドレス空間中の異なる領域ごとのアクセス制御が可能となる。

【0027】

【実施例】

【実施例1】以下、本発明の一実施例を図面に従い説明する。

【0028】図1は、同実施例のメモリ管理装置の全体構成を示すものである。この場合、メモリ管理装置1は、アドレス変換装置2、TLB(Translation Look-aside Buffer)チェック装置3、スレッド番号記憶部4、ORゲート5からなるProtection fault信号生成回路から構成されている。

【0029】そして、メモリ管理装置1は、図示しないCPUから論理ページ番号、すなわち論理アドレスの上位20ビット11、現在実行中の命令が存在する論理アドレスつまり、一命令前のプログラムカウンタPC（以下LPCと略す）の上位20ビット12、CPUのメモリアccessのモードの3ビット13を取り込み、物理ページ番号、すなわち物理アドレスの上位20ビット14を出力するようにしている。この場合、現在実行しているスレッド番号をスレッド番号記憶部4に保存するとともに、論理アドレスの上位20ビット11、LPCの上位20ビット12、CPUのメモリアccessモードの3ビット13およびスレッド番号記憶部4に記憶されている現在実行しているスレッド番号が、アドレス変換装置2とTLBチェック装置3に同時に与えられ、2つのモジュールが同じアドレス変換を並行して行なうようになる。

【0030】ここで、アドレス変換装置2は、ページテーブル（主記憶上に存在するアドレス変換テーブル）を使って論理ページ番号を物理ページ番号に変換する。一方、TLBチェック装置3は、TLB内にキャッシュされている論理アドレスと物理アドレスの対応の中に目的とする論理アドレスが存在するかを調べ、見つかった時にはそれに対応する物理ページ番号を出力するとともに、アドレス変換装置2に対してTLBヒットを16を送る。

【0031】そして、アドレス変換装置2がTLBヒット16を受けとった時は、論理ページ番号を物理ページ番号に変換するアドレス変換操作を中止する。また、TLBチェック装置3が、目的とする論理ページ番号をTLB内で見つけられなかった時は、TLBmissの信号線

## 8

17をアサートし、アドレス変換装置2によって変換が行なわれるのを待ち、最終段である第3段目のアドレス変換テーブルのエントリが見つかったところで、その内容をTLBに保存し、その中から物理ページ番号を出力する。アドレス変換装置2において、目的の論理アドレスに対応するページテーブルエントリが存在しないことが判明した場合は、ページフォルト18を発生してCPUに割り込む。また、両モジュールのいずれかにおいて、アドレス変換の最中に、そのアドレスに対するアクセスが許されないということが判明した時には、プロテクションフォルト19を発生しCPUに割り込む。

【0032】図2は、TLBチェック装置3の概略構成を示すものである。この場合、TLB31のエントリの構成要素は、論理ページ番号311、物理ページ番号312、ページの属性を示す5個のフラグ313、314、315、316、317と、3組のスレッド番号318、LPC上位20ビット319、これらスレッドに対して許されるアクセスの種類であるRWX（Rは読み込み権、Wは書き込み権、Xは実行権を示す）のパーミッションの組からなっている。以下の説明では、スレッド番号318、LPC上位20ビット319、RWXのパーミッションの組を、アクセス・コントロール・リスト、略してACLと呼ぶことにする。このACLは、どのプログラムがどの論理ページ上のコードを実行している時に、これからアクセスしようとする論理ページに対してどのようなアクセスが許されているかということを示すものである。

【0033】この場合、TLBエントリにはさらにフラグ領域があり、フラグ(V)313はTLBのエントリが有効であるか否かを示すバリッドフラグ、フラグ

(D)314はページが変更されたかを示すダーティフラグ、フラグ(R)315はページがアクセスされたかを示すリファレンスフラグ、フラグ(M)316はページに対するACLがTLB内に存在する3組以外に存在するか否かを示すモアフラグ、フラグ(E)317はページのACLのチェックを行なうかどうかを示すイネーブルフラグEからなっている。また、ACLは、スレッド番号318、LPC上位20ビット319に対して3個のフラグ320、321、322を持っている。この場合、フラグ(PE)320はスレッド番号の比較を行なうか否かを示し、フラグ(TE)321はLPC上位20ビットによる比較を行なうか否かを示し、フラグ(B)322はスレッド番号の比較を行なう際にスレッド番号の一致のみを調べるか、あるいはスレッド番号の大小関係を調べるかを示している。

【0034】しかして、外部からの論理ページ番号の20ビット11によりTLB31のエントリを特定すると、スレッド番号の20ビット15、LPCの上位20ビット12、CPUのアクセスモードの3ビット13が連結器32、33で連結され、43ビットの出力331

が得られ、これとTLB31内にキャッシュされているACLとの比較が3個の比較器34、35、36を使って行われるようになる。ここで、LPCの上位20ビット12は、後述する仮想空間の領域の領域番号の一例である。

【0035】図3は、比較器34、35、36の概略構成を示している。この場合、比較器34、35、36は、まったく同じものなので、ここでは比較器34についてのみ述べると、加算器341、342、アンドゲート343～346、オアゲート347、348、ノアゲート349、350から構成されている。

【0036】そして、このような比較器34は、ACLのフラグ(PE)320を0にすることで、スレッド番号によるアクセス条件のチェックを省くことができ、全てのスレッドに共通のアクセス許可を与えることができるようにしている。また、フラグ(TE)321を0にした場合は、LPCの値によるアクセス条件の判定を行わず、ACLのエントリに書かれているスレッド番号をもつスレッドは、LPCの値に関わらず、そのページへのアクセス許可を与えることができるようにしている。また、フラグ(B)322を1にした場合は、現在実行されているスレッド番号がACLのプログラム番号よりも大きい等しい時に、スレッド番号によるアクセス条件を満たしているものとし、フラグ(B)322が0の時は、二つのスレッド番号が等しい時に限り、スレッド番号によるアクセス条件を満たすようにしている。また、フラグ(B)322を1にすることにより、スレッド番号をリング保護におけるレベルとして使い、スレッド番号が大きいスレッドに対してより強い権利を与えるようにしている。

【0037】そして、比較器32は、2本の出力信号線351、352を有している。ここで、出力信号線352の出力は、現在実行されているスレッドのスレッド番号とACL内に書かれているスレッド番号がマッチし、且つ一つ前のプログラムカウンタPC(LPC)の上位20ビットがACL内のプログラムカウンタLPCの上位20ビットの値にマッチした場合にアサートされる。すなわち、CPUによるメモリアクセスの条件がこのACLの条件と一致していることを示すようにしている。また、出力信号線351の出力は、出力信号線352の出力がアサートされ、且つCPUによるメモリアクセスが、ACLによって許可されている種類のものではあった時にアサートされる。つまり、出力信号線351の出力がアサートされれば、メモリアクセスは許可されることになる。

【0038】そして、図2に戻って、各比較器34、35、36の出力信号線351の出力の論理和をとった結果が1であれば、TLBエントリ内の3つのACLのうちの一つによってアクセスが許されたことになるので、物理ページ番号14をイネーブルし、TLBヒット信号

線16がアサートされる。また、各比較器34、35、36の出力信号線352の出力の論理和をとった結果が1であり、出力信号線351の出力の論理和をとった結果が0の場合は、TLBエントリ内の3つのACLのうち、現在のメモリアクセスの条件に一致するものが存在するが、CPUによるメモリアクセスのモードがこのACLによって許されていない種類のものではあったことを示している。この場合には、信号線38がアサートされ、プロテクションフォルト191がアサートされる。

【0039】また、比較器34、35、36の出力信号線352の出力の論理和をとった結果が0であり、TLBエントリのモアフラグ316が0の場合は、現在のメモリアクセス条件に一致するACLが存在しないことを示している。この場合は信号線40がアサートされ、プロテクションフォルト191がアサートされる。

【0040】TLBエントリのイネーブルフラグ317の値が0のときは、信号線41がアサートされTLBヒット信号線16がアサートされる。よってACLのチェックに関係なく物理ページ番号14を出力する。また、TLBがヒットしなかった場合には、前述のようにTLBmiss信号線17がアサートされる。同時に動作しているアドレス変換装置2によって物理ページ番号が判明する可能性が残されているので、その結果を待つと同時にACLがチェックされる。

【0041】さらにTLBにはヒットしたがACLのエントリにはヒットせず、かつモアビット(M)316が1の場合はACLmiss信号線20がアサートされる。同時に動作しているアドレス変換装置2によってACLがチェックされる。次に、図4はアドレス変換装置2の概略構成を示している。

【0042】この場合、第1段目のアドレス変換装置21、第2段目のアドレス変換装置22、第3段目のアドレス変換装置23により構成されている。そして、このようなアドレス変換装置2は、スレッド番号の20ビット15、LPCの上位20ビット12、CPUのアクセスモードの3ビット13と論理ページ番号の20ビット11を取り込み、論理ページ番号の20ビット11に対応する物理ページのページテーブルエントリ24を出力するようにしている。また、TLBヒット16、プロテクションフォルト191を受け取った場合はアドレス変換を中止するようにもしている。さらに、アドレス変換の途中で、目的とする論理ページに対応する物理ページが存在するが、アクセスのモードが許されない種類のものであることが判明した場合にはプロテクションフォルト192を出力し、アドレス変換の途中で目的とする論理ページに対応する物理ページが存在しないことが判明した場合にはページフォルト18を出力するようにしている。第1段目のアドレス変換装置21では、ルートポインタ25と論理ページ番号の上位7ビット111を使って第1段ページテーブルのエントリを決定する。

【0043】第2段アドレス変換装置22では、第1段アドレス変換装置21によって判明した第2段ページテーブルの先頭アドレス26と論理ページ番号の上位8ビット目から上位13ビット目までの7ビットによって示されるページテーブル内のエントリ番号112を使って第2段ページテーブルのエントリを決定する。

【0044】第3段アドレス変換装置23では、第2段アドレス変換装置22によって判明した第3段ページテーブルの先頭アドレス27と論理ページ番号の下位6ビットで示されるページテーブル内のエントリ番号113

を使って第3段ページテーブルのエントリを決定する。図5は、第1段目アドレス変換装置21および第2段目のアドレス変換装置22の概略構成を示している。

【0045】この場合、アドレス変換装置21、22は、スレッド番号の20ビット15、LPCの上位20ビット12、CPUのアクセスモードの3ビット13を連結させた43ビットの信号線211、TLBヒット16、論理ページ番号の連続する7ビットの部分111

(112)、ルートポインタ25（先頭アドレス26）を取り込み、このうちの論理ページ番号の連続する7ビットの部分111（112）、ルートポインタ25（先頭アドレス26）をアドレス合成器212で合成することにより、この論理ページ番号に対するページテーブルエントリ213のアドレスを得るようにしている。

【0046】ページテーブルエントリ213は、次レベルページテーブルへのポインタ2131、3種類のフラグ2132～2134、3組のACL、そのページテーブルエントリに対するACLが3組を越える場合に残りのACLの置かれている場所を指すポインタ2135によって構成されている。

【0047】ここで、3種類のフラグ2132～2134のうちバリッドフラグ(V)2132は、ページテーブルエントリが有効であることを示している。モアフラグ(M)2133は、そのページに対応するACLがページテーブルエントリ内に存在する3組以外に存在するか否かを示している。イネーブルフラグ(E)2134は、ACLのチェックを行なうか否かを示している。

【0048】ページテーブルエントリの位置が決まると、バリッドフラグ2132の内容を調べ、この値が0であれば、ページフォルト18を出力し、アドレス変換中止装置227を起動し、アドレス変換を中止する。

【0049】また、43ビットの信号線211の入力と3個のACLの内容214、215、216を3つの比較器217、218、219を使って同時に比較し、この比較の結果によりアクセスが許可されることが判明した場合は、信号220がアサートされ、次レベルページテーブルへのポインタ2131の値がイネーブルされる。また、アクセスが不許可であることが判明した場合は、信号221がアサートされプロテクションフォルト192が出力される。

【0050】一方、現在のアクセス条件に一致するACLが3つのものの中から見つからなかった場合は、モアフラグ2133を調べる。そして、モアフラグ2133が0であった場合には、ACLが他にないことを示しているので、アクセスの許可がなされないことになり、信号223がアサートされ、プロテクションフォルト192を出力するようになる。また、モアフラグ2133が1であった場合は他にもACLが存在していることになるので、信号224がアサートされ、ACL切替装置225を使って、次ACLへのポインタ2135によって指示されている次ページテーブルエントリ226のACLの組に切替えて再び比較を行なうようになる。

【0051】この場合、ポインタ2135によって指示されているACLの組の中に現在のアクセス条件に一致するものがさらになかった場合には、モアビット2262が1であるかを調べ、1であれば、再びACL切替装置225によって次ACLへのポインタ2261によって指される次ページテーブルエントリのACLの組に切替えて再比較するようになる。なお、TLBヒット16を受け取った場合はアドレス変換中止装置227により変換処理を中止するようになる。図6は、第3段目のアドレス変換装置23の概略構成を示している。

【0052】この場合、アドレス変換装置23は、スレッド番号の20ビット15、LPCの上位20ビット12、CPUのアクセスモードの3ビット13を連結させた43ビットの信号線231、TLBヒット16、論理ページ番号の下位6ビットの部分113、先頭アドレス27を取り込み、このうちの論理ページ番号の下位6ビットの部分113と先頭アドレス27とをアドレス合成器232で合成することにより、この論理ページ番号に対応するページテーブルエントリ233のアドレスを得るようにしている。

【0053】ページテーブルエントリ233は、物理ページ番号2331、5種類のフラグ2332～2336、3組のACL、そのページテーブルエントリに対するACLが3組を越える場合に残りのACLの置かれている場所を指すポインタ2337によって構成されている。

【0054】ここで、3種類のフラグ2132～2134のうちバリッドフラグ(V)2332はページテーブルエントリが有効であることを示している。ダーティフラグ(D)2333はページが変更されたか否かを示している。リファレンスドフラグ(R)2334はページに対してアクセスがなされたか否かを示している。モアフラグ(M)2335はページに対応するACLがページテーブルエントリ内に存在する3組以外に存在するか否かを示している。そして、イネーブルフラグ(E)2336はACLのチェックを行なうか否かを示している。

【0055】ページテーブルエントリの位置が決まると、バリッドフラグ2332の内容を調べ、この値が0

であれば、ページフォルト18を出力し、アドレス変換中止装置247を起動しアドレス変換を中止する。

【0056】また、43ビットの信号線231の入力と3個のACLの内容234、235、236を3つの比較器237、238、239を使って同時に比較し、この比較の結果によりアクセスが許可されることが判明した場合は、信号240がアサートされ、物理ページ番号2331とフラグ2332～2336の組と3つのACLの内容234～236を連結した192ビットの出力2338がイネーブルされる。また、アクセスが不許可であることが判明した場合は、信号241がアサートされ、プロテクションフォルト192が出力される。

【0057】一方、現在のアクセス条件に一致するACLが3つのものの中から見つからなかった場合は、モアフラグ2335を調べる。モアフラグ2335が0であった場合には、ACLが他にはないことを示しているので、アクセスの許可がなされないことになり、信号242がアサートされ、プロテクションフォルト192を出力するようになる。また、モアフラグ2335が1であった場合は他にもACLが存在していることになるので、信号243がアサートされ、ACL切替装置245を使って、次ACLへのポインタ2337によって指示されている次ページテーブルエントリ246のACLの組に切替えて、再び比較を行なう。

【0058】この場合、ポインタ2135によって指示されているACLの組の中に現在のアクセス条件に一致するものがさらになかった場合には、モアビット2262が1であるかを調べ、1であれば、再びACL切替装置245によって次ACLへのポインタ2461によって指されるACLの組に切替えて再び比較を行なうようになる。

【0059】図7は、ACLの組を一回切替えた後の比較によってアクセスが許可されることが判明した場合の状態を示すもので、図6と同一部分には同符号を付して示している。この場合、ページテーブルエントリ233の物理ページ番号2331とフラグ2332～2336の組とページテーブルエントリ246の3つのACLの内容247～249を連結した192ビットの出力2462がイネーブルされるようになる。

【0060】本実施例では、メモリ管理装置中のページテーブルのエントリ中に記載するACLはLPCの上位20ビット固定であるが、この場合、アクセス元の領域の大きさが固定される。領域の大きさを可変または複数の領域をまとめて1つのACLで表現できれば柔軟性が増す。その場合には図12のように1つのACL中にさらにマスク領域322（最大でLPC上位ビット数と同じビット数）を設け、このマスクに従って比較を行うようにすればよい。ACL中のスレッド番号についても同様なマスク領域をさらに付加すれば複数のスレッドをグループでまとめたアクセス制御が可能となる。

【0061】また、図3においてアクセス権の検証を行う比較器の構成でスレッド番号及びLPCの比較を行うのに、減算器341、342を用いているが、ここを論理演算器に置きかえてもよい。この場合、一例として減算器342のかわりにAND演算器とする。そしてアドレス空間の上位をOSなどのシステムに、下位をアプリケーションのプログラムに配置する構成とする。LPC上位20ビット319に16進数で80000が入っているとLPC上位20ビット12の値が16進数で80000以上の場合、AND演算器の出力は1になる。つまり、システムプログラムからのアクセスは許可され、アプリケーションのプログラムからのアクセスは拒否される。このように種々の論理演算器及びその組合せに置きかえることが可能になる。

【0062】また、本実施例では、論理アドレスから物理アドレスを求めるのに3段のアドレス変換装置を利用し、かつ、各段においてACLのチェックを行っているが、最終段のみACLチェックを行い第1段、第2段においてはACLのチェックを行わない構成も取り得る。

#### 【実施例2】

【0063】以上第1の実施例では、従来の第1、第2、第3の問題点を解決する手段をまとめて示したが、ここで、上記実施例のうち、従来の第1の問題点を解決する手段を取り出し、第2の実施例として説明する。まずアクセス権限の判定に関して説明する。

【0064】図8において、論理アドレス空間80の中には、複数のスレッド81、82が存在している。ページテーブル85は図2の一部を取り出したものである。スレッド番号1を持つスレッド81が、あるアドレスのメモリをアクセスする命令83を実行しようとする。ここで命令83は、16進数表記で1234番地のデータをロードしてくることを意味する。まず、ページテーブル85の中から、アクセスしようとするアドレスが存在するページに対応するエントリが選ばれる。次に、スレッド番号検出部86によって、現在実行中のスレッド番号が検出され、このスレッド番号に一致するスレッド番号を持った保護情報のエントリが選択される。

【0065】このエントリの中に書いてある、そのスレッドに許されるアクセスの種類は、アクセスの種類検出部84に入力される。84は、実行されつつあるアクセスの種類が、スレッドに許されるアクセスの種類に含まれるか否かを検出し、その結果を物理アドレス／プロテクション・フォルト信号出力部87に送る。87は、アクセスが許される場合は、アクセスされようとしている論理ページの番号に対応する物理ページの番号と、論理ページ内のオフセットを合わせて物理アドレスとして出力する。アクセスが許されない場合は、プロテクションフォルト信号を出力する。

【0066】図8のページテーブル85の内容によると、論理ページ番号が1であるページは、スレッド1か



らは読み書きが出来るが実行は出来ず、スレッド2からは読み書きが出来ないが実行は出来る、というようになっている。このように、一つのアドレス空間内を走る複数のスレッドについて、別々のアクセス権限を与えることが出来る。

【0067】さらに詳しく説明すると、スレッドがある論理ページをアクセスしようとする、ページテーブルの中の、前記論理ページに対応するエントリが選択され、まず、前記論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かを、フラグによってページフォルト検出部88が検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【0068】それ以外の場合は、そのエントリの中に複数存在する、アクセスを許すスレッドの番号のフィールドのうちで、現在実行中のスレッド番号と一致するものが存在するかどうかを、比較器を使って検出する。比較器を複数用意して並列に比較すると、比較のための時間が一回比較する時間で済むことになる。

【0069】一致するフィールドが存在しない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致するフィールドがあった場合は、スレッドが行なおうとしているアクセスの種類が、そのスレッドに許されているアクセスの種類に含まれているかどうかを、比較器を用いて検出する。この比較は、スレッド番号の一致を検出することと同時に進行することで比較時間を短縮することが出来る。含まれていることが検出されなければプロテクションフォルトとして、割り込み信号を出力する。含まれていることが検出されたときは、アクセスされつつある論理ページに対応する物理ページ番号を出力する。

【0070】OSがスケジューリングによって、一つのアドレス空間の中で実行するスレッドを切り替える場合は、メモリ管理装置内のスレッド番号記憶部に保存されている、実行中のスレッドの番号を変更する。これにより、これ以降のアドレス変換時は、変更後のスレッド番号と、その論理ページをアクセス出来るスレッド番号の一致を調べることになる。一つのアドレス空間の中を実行する複数のスレッドに関するページテーブルは、一つにまとまっているため、OSがページングを行なう時は、前記ページテーブルの該当するエントリの、論理-物理アドレスの対応の部分の、物理アドレスの部分を書き換え、そのエントリに対応する物理ページが存在するかどうかを示すフラグを更新するだけでよい。

【実施例3】さらに、第1の実施例のうち、従来の第3の問題点を解決する手段を取り出し、第3の実施例として説明する。まず、アクセス権限の判定に関して説明する。

【0071】図9において、論理アドレス空間90は、複数の領域91、92、93に分割されている。ページ

テーブル96は図2の一部を取り出したものである。領域91の中のプログラムを実行中に、あるアドレスのメモリをアクセスする命令94を実行しようとする、まず、ページテーブル96の中から、アクセスしようとするアドレスの存在するページに対応するエントリが選ばれる。次に、領域番号検出部97によって、現在実行されつつあるプログラムの存在する領域の領域番号を検出し、この領域番号に一致する領域番号を持った保護情報のエントリが選択される。

【0072】このエントリの中に書いてある、その領域内のプログラムに対して許されるアクセスの種類は、アクセスの種類検出部95に入力される。95は、実行されつつあるアクセスの種類が、その領域内のプログラムに対して許されるアクセスの種類に含まれるか否かを検出し、その結果を物理アドレス/プロテクション・フォルト信号出力部98に送る。98は、アクセスが許される場合は、アクセスされようとしている論理ページの番号に対応する物理ページの番号と、論理ページ内のオフセットを合わせて物理アドレスとして出力する。アクセスが許されない場合は、プロテクションフォルト信号を出力する。

【0073】図9のページテーブル96の内容によると、論理ページ番号が10であるページは、領域1からは読み書きは出来ないが実行は出来、領域2からは読み書きは出来るが実行は出来ない、となっている。このように、一つの論理アドレス空間内の複数のプログラムに、別々のアクセス権限を与えることが出来る。

【0074】なお、論理アドレス空間を分割する領域の大きさは、1バイト単位、ページ単位、1メガバイト単位など、どのような単位でも構わない。また、ページの大きさも、好きなように決めても構わない。

【0075】さらに詳しく説明すると、スレッドが実行しているプログラムが置かれている領域によって、アクセス権限を変える場合は、実行中の命令の存在するアドレスつまり1つ前のプログラムカウンタの上位何ビットかを、現在実行されているプログラムの領域の番号として使用する。

【0076】まず、アクセスされようとしている論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かを、フラグによってページフォルト検出部99で検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【0077】次に、アクセスを行なうプログラムが置かれている領域の領域番号と、ページテーブル中の論理-物理アドレスの各対応毎に記憶されている、アクセスを許す領域の番号が一致するかどうかを、比較器を使って検出する。一致する領域番号がない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致する番号があった場合は、これから実行されようとしてい

るアクセスの種類が、その領域から許されるアクセスの種類に含まれているかどうかを比較器を用いて検出する。含まれていることが検出されなければプロテクションフォルトとして割り込み信号を出力する。含まれていることが検出された時は、アクセスされつつある論理ページに対応する物理ページの番号を出力する。

【実施例4】さらに第1の実施例のうち、従来の第3の問題点を解決する部分の変形例を第4の実施例として説明する。まず、アクセス権限の判定に関して説明する。

【0078】図10において、論理アドレス空間100は、複数の領域101、102、103に分割されている。領域101の中のプログラムを実行中に、あるアドレスのメモリをアクセスする命令104を実行しようすると、領域番号検出部108によって現在実行されつつあるプログラムの存在する領域の領域番号を検出する。次に、その領域番号を利用してACLテーブル107の対応する領域番号のACLEントリを選択する。次にアクセスしようとするアドレスの存在するページ番号を持った保護情報のエントリが選択される。

【0079】このエントリの中に書いてあるアクセス先ページに対して許されるアクセスの種類はアクセスの種類検出部105に入力される。105は実行されつつあるアクセスの種類が、そのアクセス先のページに対して許されるアクセスの種類に含まれるか否かを検出してもし許されない場合はプロテクションフォルト信号を出力する。

【0080】図10のACLテーブル107の内容によると、領域番号が1である領域からは論理ページ番号10へは読み書きはできないが実行のみ出来、論理ページ番号11へは読み書きができるが、実行は出来ない、となっている。このように、一つの論理アドレス空間内の複数のプログラムに、別々のアクセス権限を与えることが出来る。

【0081】なお、論理アドレス空間を分割する領域の大きさは、1バイト単位、ページ単位、1メガバイト単位など、どのような単位でも構わない。また、ページの大きさも、好きなように決めても構わない。

【0082】更に詳しく説明すると、スレッドが実行しているプログラムが置かれている領域によって、アクセス権限を変える場合は、実行中の命令の存在するアドレス、つまり1つ前のプログラムカウンタの上位何ビットかを、現在実行されているプログラムの領域の番号として使用する。

【0083】まず、アクセスされようとする論理ページが有効かどうか、さらに有効な場合に物理メモリが割り当てられているか否かをページテーブル106のフラグによってページフォルト検出部110が検出し、もし無効の場合や割り当てられていない場合は、ページフォルトとして割り込み信号を発生する。

【0084】次に、アクセスを行なう先の論理ページ番

号と、ACLテーブル107中の、アクセスを行なうプログラムが置かれている領域の領域番号毎に記憶されている、アクセスを許す論理ページの番号が一致するかどうかを、比較器を使って検出する。一致する論理ページ番号がない場合は、プロテクションフォルトとして、割り込み信号を出力する。一致する番号があった場合は、これから実行されようとしているアクセスの種類が、そのページに対して許されるアクセスの種類に含まれているかどうかを比較器を用いて検出する。含まれていることが検出されなければプロテクションフォルトとして割り込み信号を出力する。プロテクションフォルト信号が出力されないことで含まれていることが検出された時は、アクセスされつつある論理ページに対応する物理ページの番号を物理アドレス出力部109から出力する。

【0085】また、図10におけるACLテーブル107中のACLEントリはアクセス先の論理ページ番号とそのページに対するアクセスの種類の対で記載されているが、アクセス先の領域番号とその領域に対するアクセスの種類の対で記載してもよい。その場合、図11のようになりACLテーブル107が112のように変更され、新たに設ける領域番号検出部111によって、アクセス先のアドレスに相当する領域番号を検出して、ACLテーブル112の対応するエントリを選択する。あとの処理は図10の場合と同様である。

【0086】なお、第3の実施例では領域2中の論理アドレスに対して領域1中のプログラムからアクセスできるかという検証方法を、本実施例では領域1中のプログラムを実行中に領域2中の論理アドレスに対してアクセスできるかという検証方法を採用しているが、ページテーブルをハッシュを用いて構成することにより両者を融合した検証方法も採り得る。また、本実施例でページテーブル106とACLテーブル107を別々に設けているのは、メモリ容量の節約のためである。

【0087】このように上記第1～第4の実施例によれば、論理アドレス空間を共有する複数のスレッドが、複数の領域におかれているプログラムを実行することによって行なうメモリアccessに対して、TLBやページテーブル中の一つのエントリにACLを複数個格納して対応させるようにしたので、同じ物理アドレスに対して多くのエントリを占有することがなく、しかも複数のエントリが論理アドレスと物理アドレスの対を重複して持つことがなくなる。従って、ページテーブル占有するメモリの量が減少する。また、TLB（ページテーブルのキャッシュ）中に格納されている論理アドレスの範囲が広くなり、TLBのヒット率を向上することができる。また、TLBやページテーブル中の複数のACLのそれぞれと、現在実行中のスレッドのスレッド番号と実行されているプログラムの置かれている領域番号の組の比較を同時に行なうことができる。このため、アドレス変換の時間は、従来のメモリ管理装置と同等にできる。

【0088】また、上記実施例によれば、TLBを検索中に発生したミス、アドレスのミスと、ACLのミスに分けることが可能となり、また、TLBのアドレスを格納する部分と、ACLを格納する部分を、独立にページテーブル中の情報と入れ換えることができる。これにより、TLBの一つのエントリ内に、そのエントリに対応する論理ページに対するACLをすべて格納できない場合でも、当該エントリを無効化せず、ACLの部分のみを入れ換えることができる。従って、TLBミスの処理にかかる時間を短縮することができる。これは特に、論理アドレスを共有するスレッドあるいはプログラムの置かれている領域の数が、一つのエントリ中に格納できる数より多い場合に有効である。

【0089】なお、上記実施例では、ページテーブルが保持する複数のACLについて、スレッド番号と領域番号の組と、その組に許可するアクセスの種類が一つづつ組み合わせさせている場合を示したが、複数のスレッド番号と領域番号の組に対して、共通に許可するアクセスの種類を一つだけ記憶するようにしても良いし、複数のスレッド番号に対して領域番号と許可するアクセスの種類を各一つ共通に記憶しても、複数の領域番号に対してスレッド番号と許可するアクセスの種類を各一つ共通に記憶しても良い。スレッドあるいは領域毎に個別に許されるアクセス方法を記憶させればきめ細かいアクセス権制御ができ、一方、アクセスを許すスレッドあるいは領域の集合に共有させて許されるアクセス方法を一つ記憶させればアクセス権制御のきめ細かさは多少落ちるが、ページテーブルとして使用されるメモリの量が節約できる。

【0090】また、上記実施例では保護の最小単位を物理ページ単位としたが、メモリ管理装置が管理できる大きさであればどのような大きさが単位であっても、またセグメントのように可変量であっても良い。

【0091】その他、TLBが保持するページテーブルの一つのエントリに複数のACLを保持するという第2の発明は、どのように情報が格納されている場合にも適用可能であることはもとより、ACLの内容として、上記のスレッド番号と、領域番号であるLPC上位20ビットの組以外でも適用可能であり、また、各信号発生回路および制御回路の構成、各種制御信号の与え方なども種々変更して実施することができる。また実施例4のようにページテーブルとACLテーブルを別々に分けることにより、全体のメモリ量を減らすこともできる。

【0092】また、上記実施例によれば、アクセスの権利の正しさを検証するため、スレッドを識別する番号だけでなく、スレッド毎のメモリアccessの仕方を示すアクセス情報のエントリを設けることが可能となり、メモリアccessに対してきめ細かいアクセス権制御を行なうことが可能になる。

【0093】また、上記実施例によれば、メモリアクセ

ス毎にスレッド番号あるいは領域番号の全部又は一部をマスクしてグループ化し一致の検出を行なうことが可能となる。これはメモリアccessに対してグループ化したアクセス権制御を行なうようにできる。そのため、効率的なアクセス権制御を行うことができる。

【0094】また、上記実施例によれば、スレッド番号あるいは領域番号の一致を検出するだけでなく、大小関係が成立する等、所定の論理演算結果が真となる場合を一致が得られたとすることができ、メモリアccessに対してプログラム間の又はスレッド間の相対的なアクセス権制御を行なうようにできる。

【0095】

【発明の効果】本発明は、以上に説明したように構成されているので、以下に記載するような効果を奏する。

【0096】まず、ページテーブル内の論理—物理アドレスの対応に対して、どのスレッドがその論理アドレスに対してアクセス可能かという情報を記憶する手段を設けることにより、一つのアドレス空間内を実行する複数のスレッド同士の保護を、ページテーブルのために使用するメモリの量を無駄にすることなく、行なうことが出来る。

【0097】更に、1つの論理—物理アドレス対に対して複数のアクセス保護情報を記憶すれば、一つの論理—物理アドレスの対応を記憶するために使われるメモリの量が少なくなるため、TLBに置くことができる論理—物理アドレスの対応の数が増え、キャッシュのヒット率が上がるため、平均のアドレス変換時間が短縮される。

【0098】そして、一つのアドレス空間を複数の領域に分割し、ページテーブル内の一つの論理—物理アドレスの対応に対して、どの領域に存在するプログラムを実行した時に、その論理アドレスに対してアクセス可能かという情報を1個以上記憶する手段を設けることにより、一つのアドレス空間内に存在する複数のプログラムの対して別々のアクセス権限を与えることが出来る。これにより、あるデータをアクセスするためには、必ず、ある特定のアクセスプログラムを介してしか行われないうにすることができる。

【0099】また、一つのアドレス空間を複数の領域に分割し、かつ複数のスレッドを実行させる場合に、ページテーブル内の論理—物理アドレスの対応に対して、どのスレッドがどの領域に置かれているプログラムを実行している時に、その論理ページに対してアクセス可能かという情報を記憶する手段を設けることにより、上で述べた効果が同時に達成される。

【0100】上記のアクセス可能か否かを示す複数の情報を、並行して検証すれば、一つの論理—物理アドレスの対応に対して複数の保護情報があっても、アクセス権のチェックは保護情報が1個しかない場合と同じ時間で行なうことができる。

【0101】尚、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号の一致を調べてアクセスの正当性を検証する際に、スレッド番号の一部をマスクする機能を具備すれば、スレッド番号のグルーピングが可能となるため、複数のスレッドに対して同じ権限を与えたい場合にページテーブルに置かなければならない情報の量とチェック時間を節約することが出来る。

【0102】同様に、検証の際に、どのスレッドがアクセス可能かを示す情報のなかのスレッド番号と、アクセスしようとしているスレッドのスレッド番号についての論理演算の結果に基づきスレッド番号が一致したと見做す機能を具備することによって、スレッド番号のグルーピングが可能となるため、複数のスレッドに対して同じ権限を与えたい場合にページテーブルに置かなければならない情報の量とチェック時間を節約することが出来る。このようなグルーピングは領域番号についても同様に実現できる。

【0103】さらに、TLB（ページテーブルのキャッシュ）を検索中に発生したミス、アドレスのミスと、ACL（保護情報）のミスに分けることが可能となり、また、TLBのアドレスを格納する部分と、ACLを格納する部分を、独立にページテーブル中の情報と入れ換えることができる。これにより、TLBの一つの論理ページに対応するエントリ内に、その論理ページに対するACLをすべて格納できない場合でも、当該エントリを無効化せず、ACLの部分のみを入れ換えることができる。従って、TLBミスの処理にかかる時間及び保護情報ミスの処理にかかる時間を短縮することができる。

【0104】このように本発明によれば、一回のアドレス変換に要する時間は従来と同等で、ページテーブルのヒット率が向上し、ミス処理にかかる時間を短縮できることから、全体としてアドレス変換時間を短縮できることになり、また、複数のスレッドが複数の領域に分割されたアドレス空間を実行する時の保護を、スレッド番号と領域番号によってOSの介在なしで柔軟に管理することが出来るようになる。

#### 【図面の簡単な説明】

【図1】 本発明のメモリ管理装置の一実施例を示す概略構成図。

【図2】 図1に示す実施例に用いられるTLBチェック装置の概略構成を示す図。

【図3】 図2に示すTLBチェック装置に用いられる比較器の概略構成を示す図。

【図4】 図1に示す実施例に用いられるアドレス変換装置の概略構成を示す図。

【図5】 図4に示すアドレス変換装置に用いられる第1段および第2段目のアドレス変換装置の概略構成を示す図。

【図6】 図4に示すアドレス変換装置に用いられる第3段目のアドレス変換装置の概略構成を示す図。

【図7】 図6に示すアドレス変換装置の動作を説明するための図。

【図8】 本発明のメモリ管理装置の第2の実施例を示す概略構成図。

10 【図9】 本発明のメモリ管理装置の第3の実施例を示す概略構成図。

【図10】 本発明のメモリ管理装置の第4の実施例を示す概略構成図。

【図11】 第4の実施例の変形例を示す概略構成図。

【図12】 スレッド番号あるいは領域番号の全部または一部をマスクする構成の一例を示す図。

#### 【符号の説明】

1…メモリ管理装置

2…アドレス変換装置

20 3…TLBチェック装置

4…スレッド番号記憶部

21、22、23…アドレス変換器

25…ルートポインタ

212、232…アドレス合成器

217～219、237～239…比較器

225、245…ACL切替え装置

227、247…アドレス変換中止装置

31…TLB

32、33…連結器

30 34、35、36…比較器

80、90、100…論理アドレス空間

81、82…スレッド

83、94、104…実行しようとしている命令

84、95、105…アクセスの種類検出部

85、96、106…ページテーブル

86…スレッド番号検出部

87、98…物理アドレス／プロテクション・フォルト信号出力部

88、99、110…ページフォルト検出部

40 91、92、93、101、102、103…領域

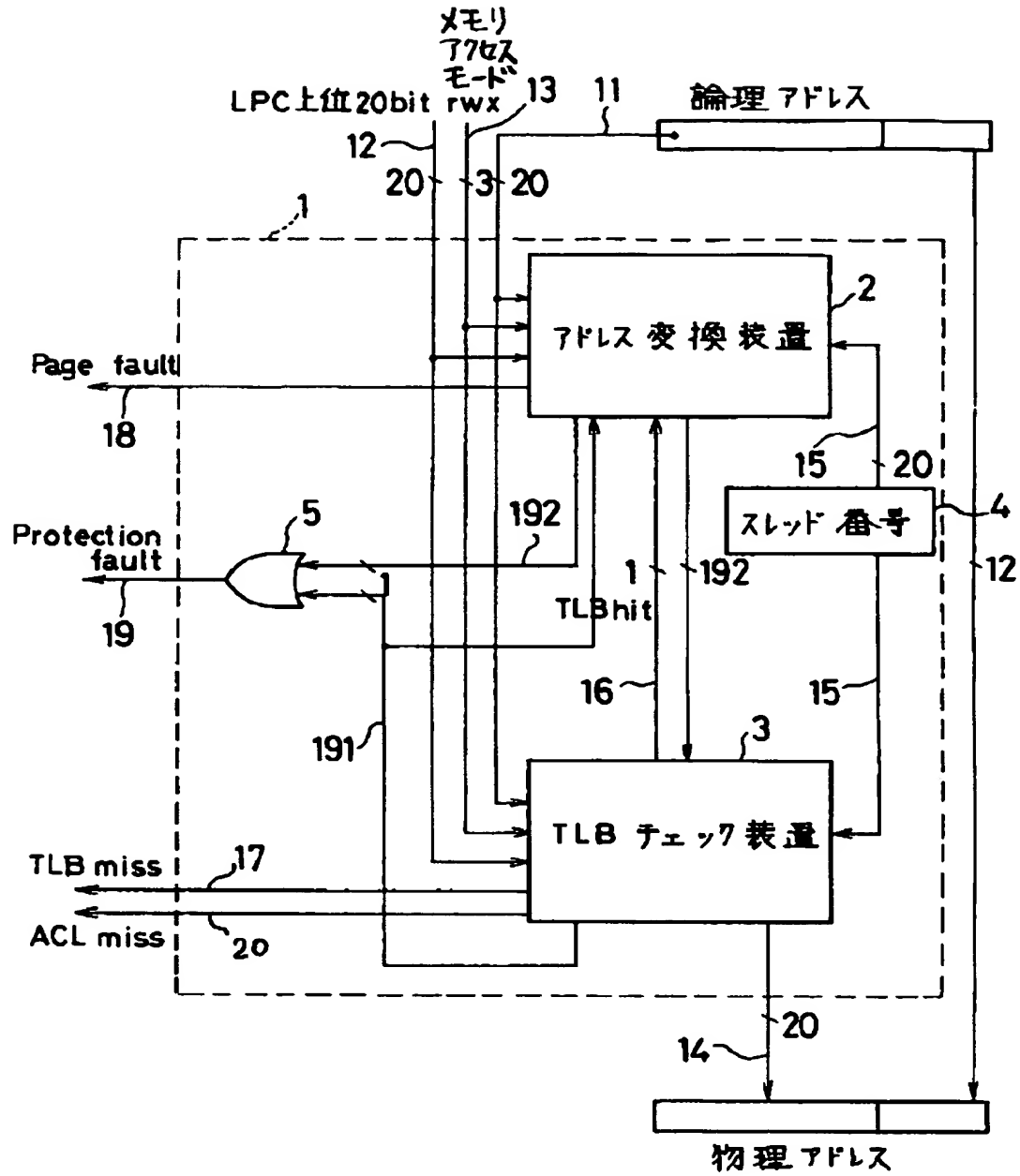
97、108…実行しようとしている命令が存在するアドレスを含む領域の番号検出部

107、112…ACLテーブル

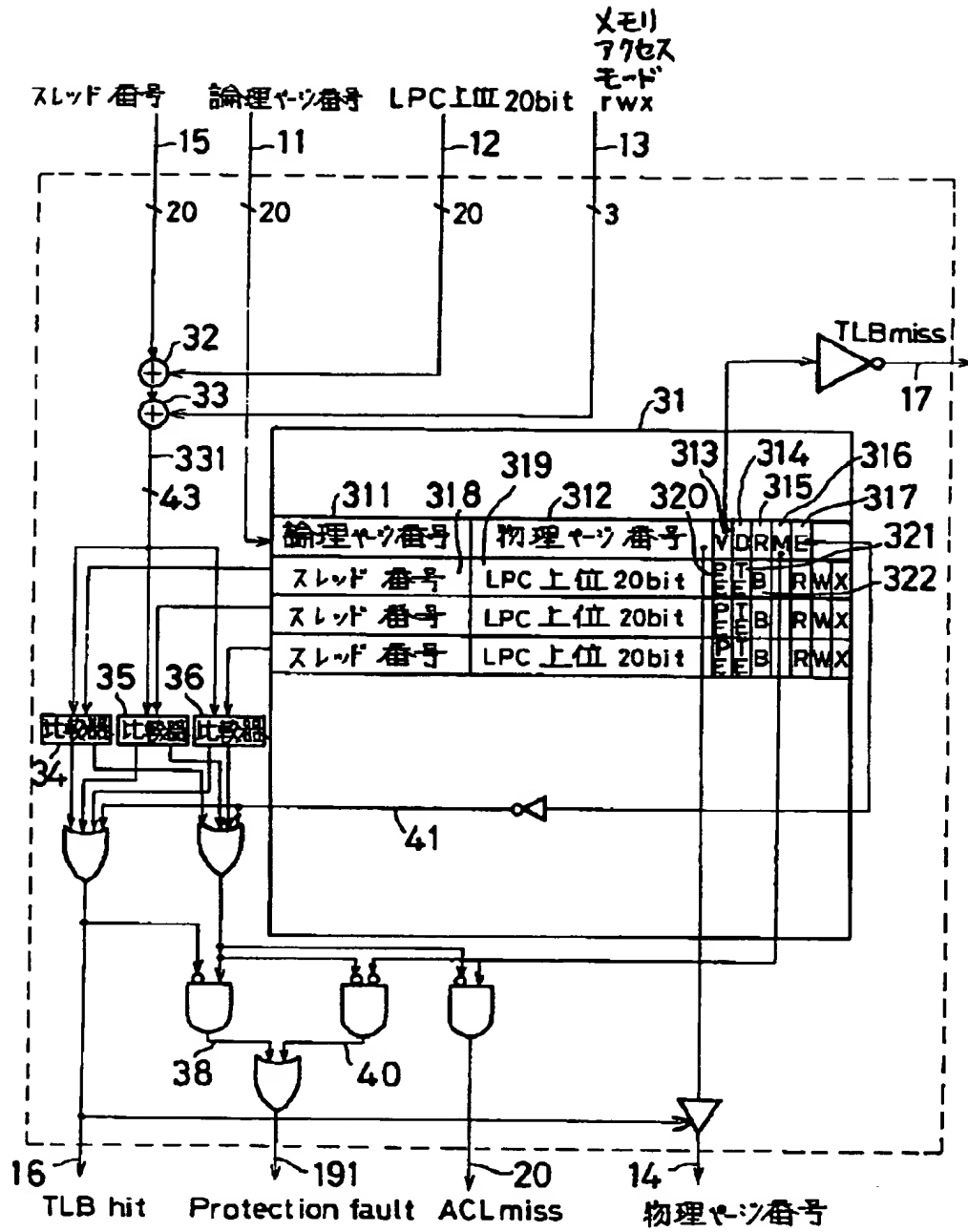
109…物理アドレス出力部

111…アクセスしようとする先のアドレスを含む領域の番号検出部

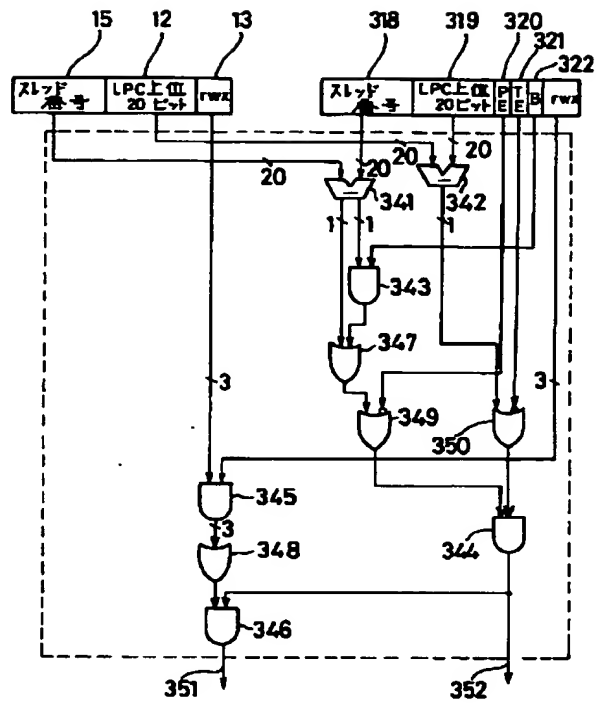
【図1】



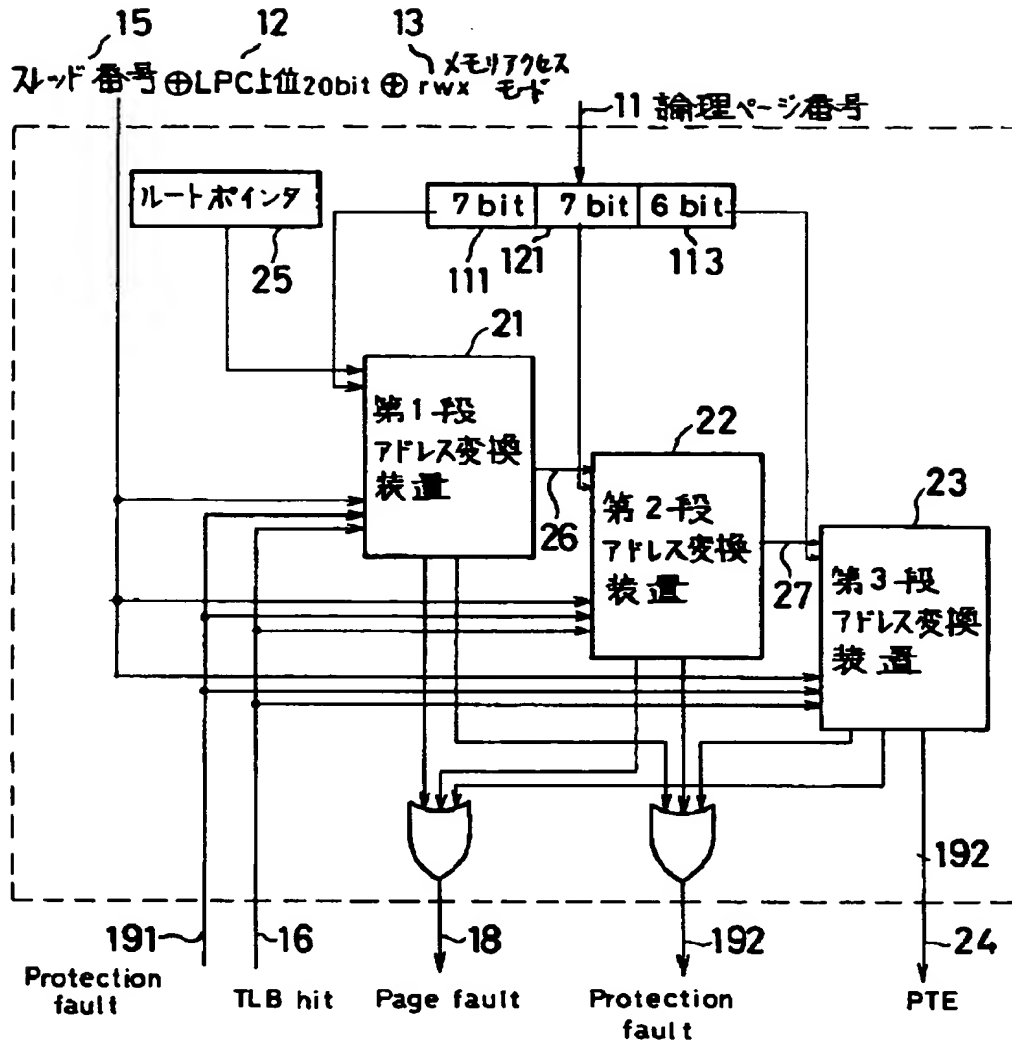
【図2】



【图 3】

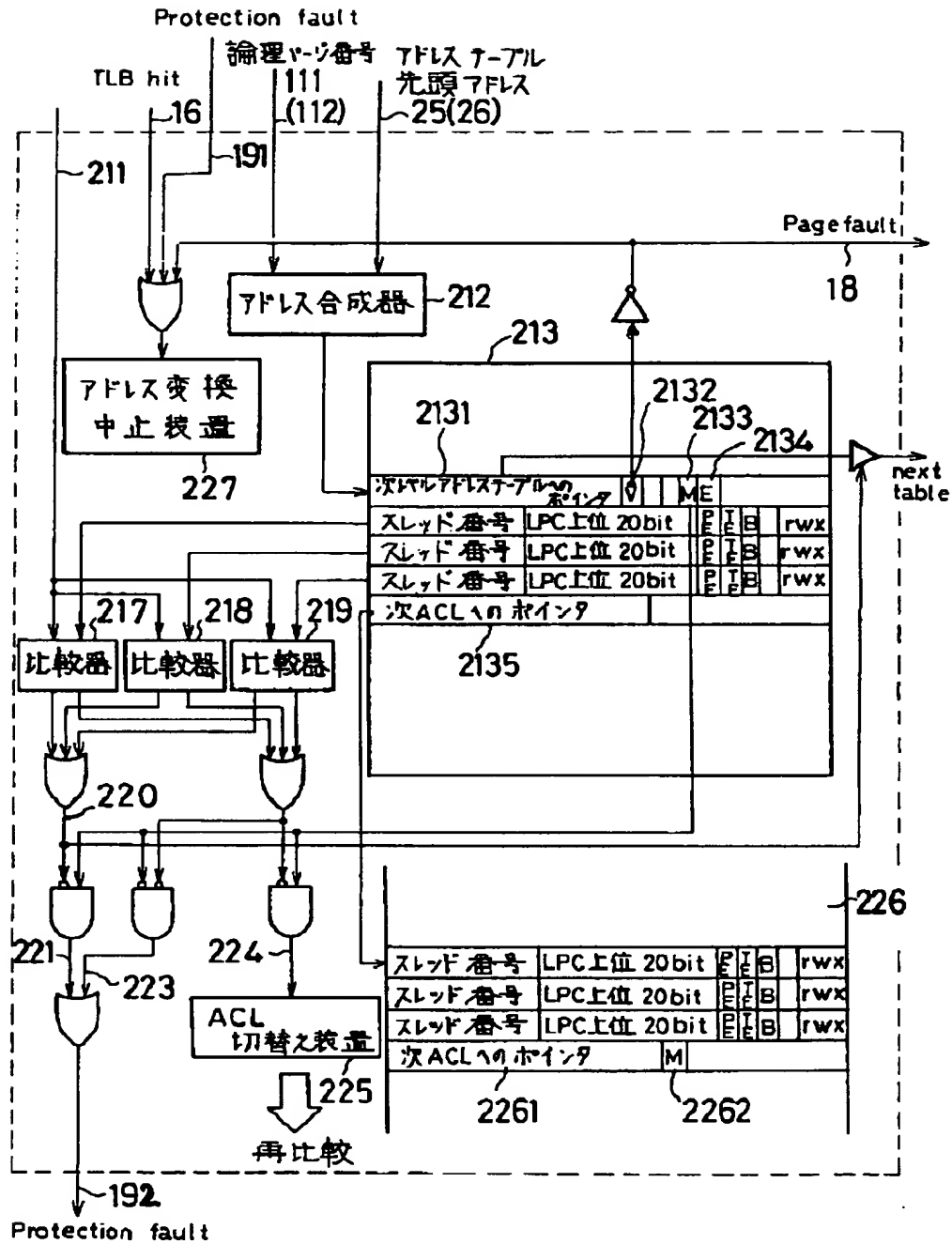


【図4】

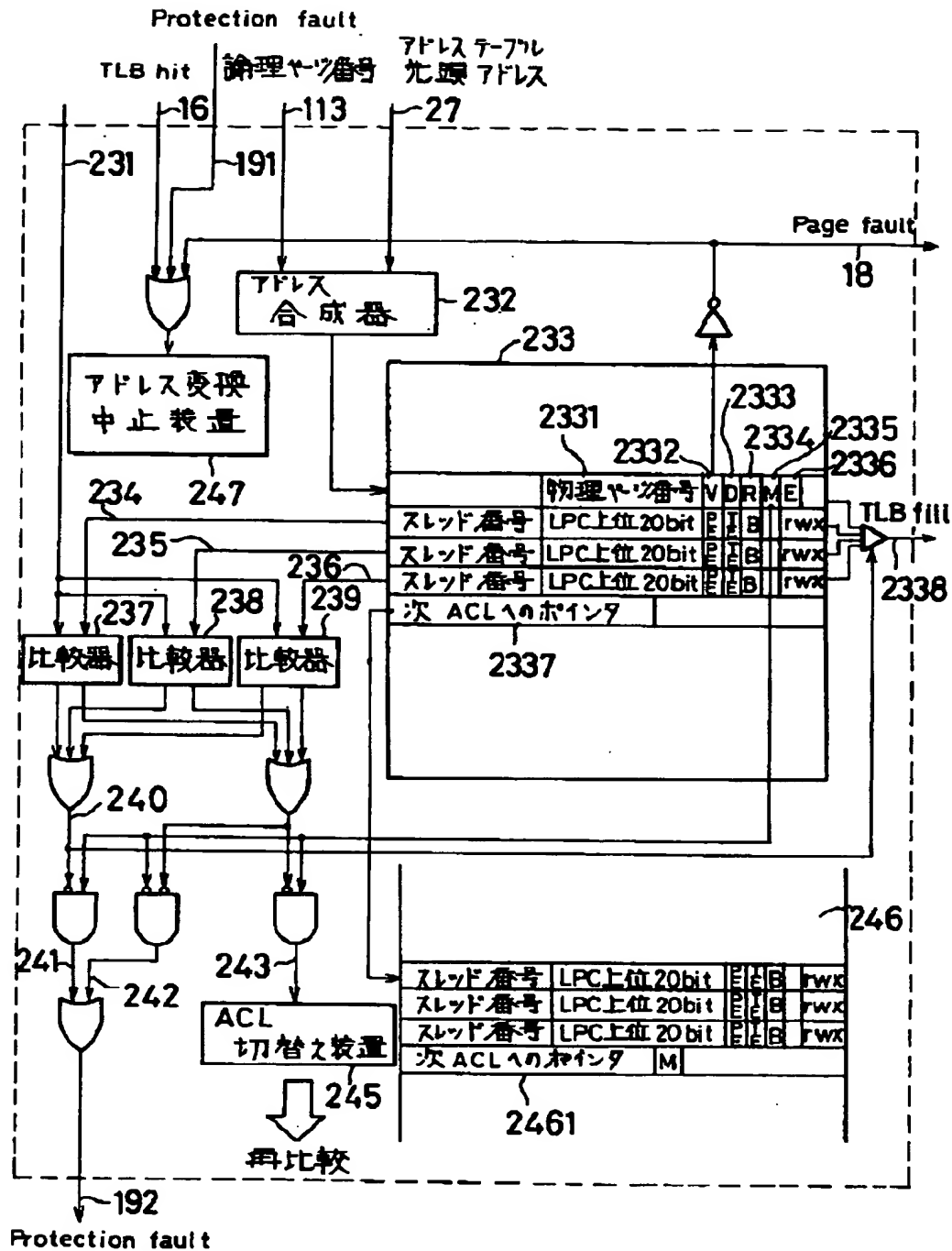




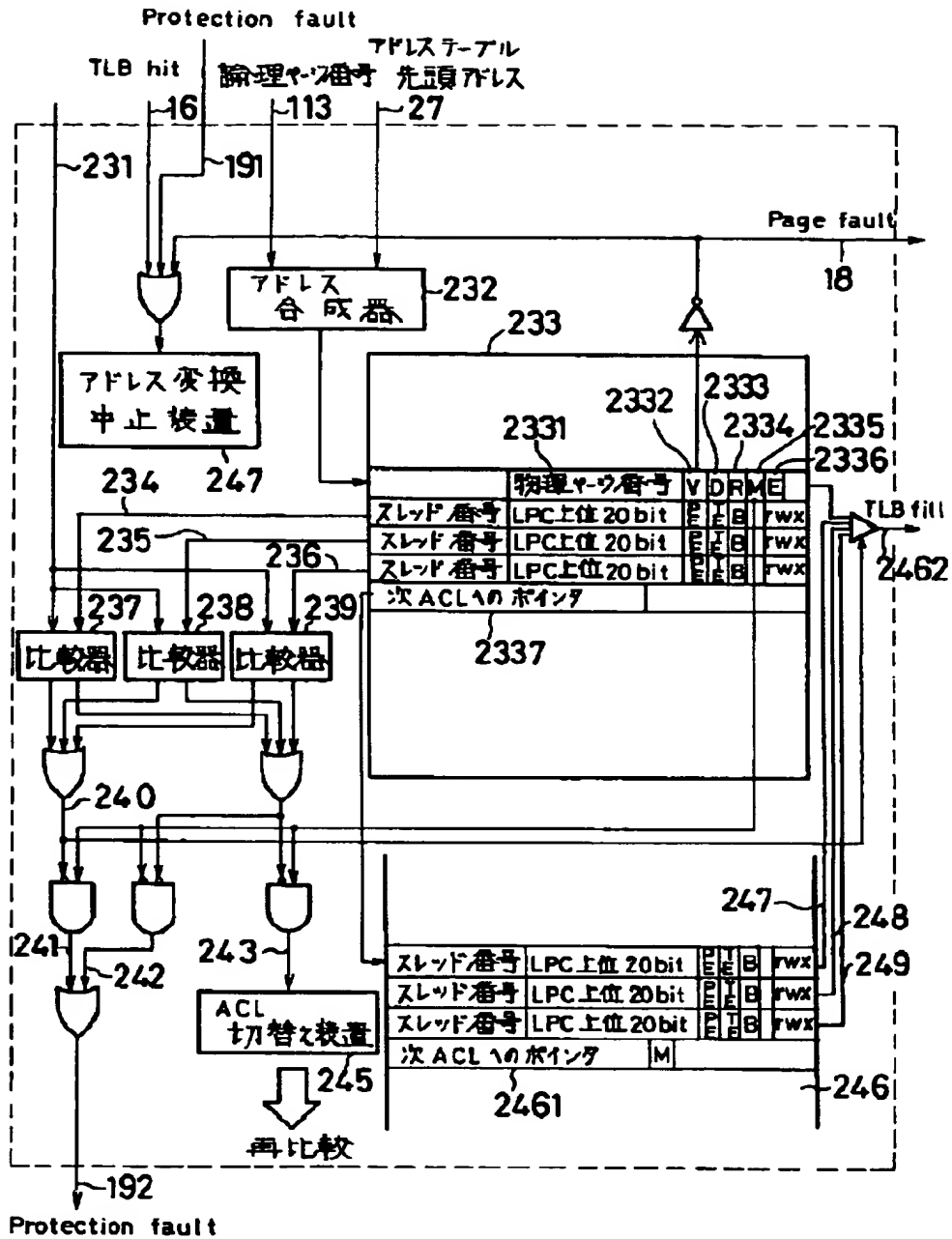
【図5】



【図6】



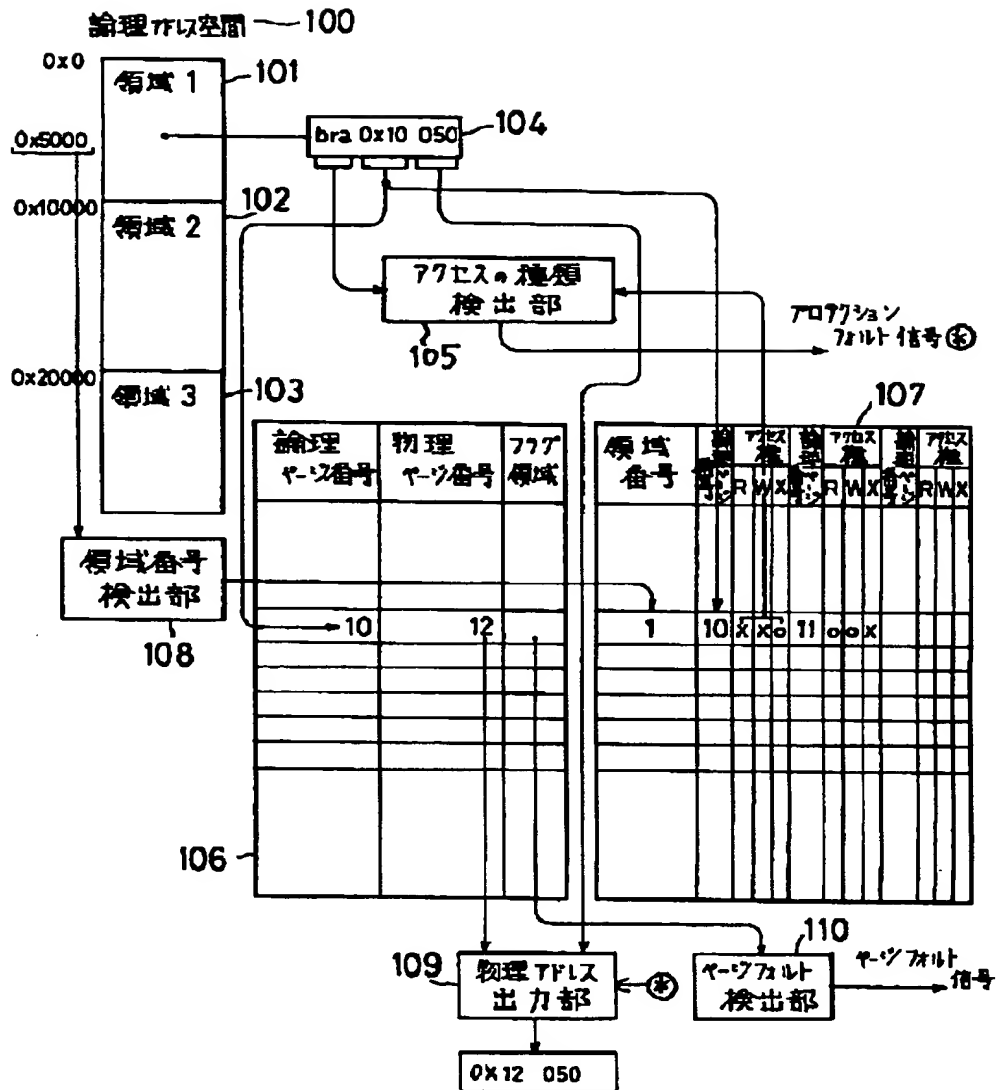
【図7】





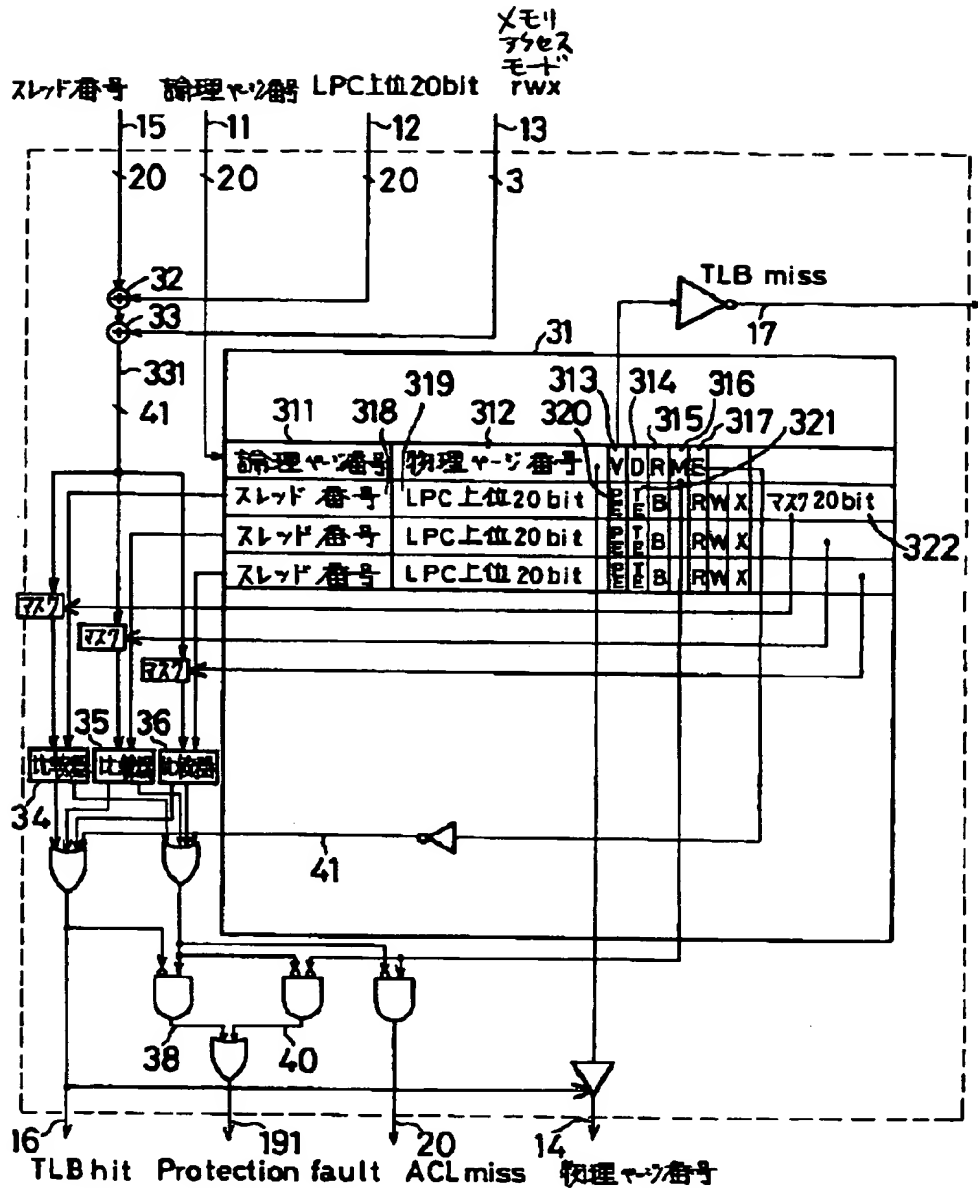


【☒ 1 0】





【図12】



フロントページの続き

(72)発明者 野末 浩志  
 神奈川県川崎市幸区小向東芝町1番地 株  
 式会社東芝研究開発センター内

(72)発明者 前田 賢一  
 神奈川県川崎市幸区小向東芝町1番地 株  
 式会社東芝研究開発センター内

(72)発明者 瀬川 英生  
 神奈川県川崎市幸区小向東芝町1番地 株  
 式会社東芝研究開発センター内

(72)発明者 岡本 利夫  
 神奈川県川崎市幸区小向東芝町1番地 株  
 式会社東芝研究開発センター内